Санкт-Петербургский государственный университет

Направление «Математическое обеспечение и администрирование информационных систем»

Профиль «Информационно-аналитические системы»

Виноградова Анастасия Александровна

Системы разделения секрета

Выпускная квалификационная работа

Научный руководитель:

д. т. н., профессор Крук Е. А.

Рецензент:

к.т.н., доцент Овчинников А. А.

Санкт-Петербург

2017

SAINT-PETERSBURG STATE UNIVERSITY

Main Field of Study «Software and Administration of Information Systems»

Area of Specialisation «Information and Analytical Systems»

Anastasiia Vinogradova

Secret sharing systems

Graduation Project

Scientific supervisor:

DrSci, professor Eugene Krouk

Reviewer:

Ph.D., associate professor Andrei Ovchinnikov

Saint-Petersburg

2017

Оглавление

Введение

Понятие секретных данных, как данных, не подлежащих разглашению, или на распространение которых наложены ограничения, известно людям с давних времен и понятно каждому. Такие данные нуждаются в защите от несогласованного разглашения и передачи, а также непредвиденной потери, для этого нужно обеспечить надежность их хранения.

Одним из способов предотвратить потерю данных является создание копий секретной информации и хранении их в разных местах, но этот же способ имеет значительный недостаток, а именно, вероятность компрометации данных существенно увеличивается. Еще одним способом является разделение данных между участниками, которые заинтересованы в сохранении секрета.

Современные данные чаще всего хранятся на электронных устройствах, таким образом, задача построения и реализации в программном виде разделения данных между участниками и последующего восстановления секрета требует всестороннего изучения.

1. Цель и постановка задачи

Целью работы является написание библиотеки для разделения секрета на одном из широко распространенных языков программирования. Такая библиотека позволяла бы разделять секретные данные, раздавать доли участникам и восстанавливать секрет. Для достижения результата были поставлены следующие задачи:

1. Изучение предметной области;
2. Выбор схемы разделения секрета;
3. Разработка библиотеки;
4. Тестирование библиотеки;
5. Сравнительный анализ с другими реализациями систем.

2. Введение в предметную область

Задача разделения секрета подразумевает под собой разделение секретной информации между участниками так, что только заранее заданные множества участников смогут ее восстановить, следственно, вероятность компрометации этой информации снижается.

Наглядным примером может послужить комната, в которой хранится нечто ценное для определенной группы лиц. Если закрыть комнату на несколько различных между собой замков, количество которых равно количеству участников, и раздать каждому члену этой группы по одному ключу, то открыть дверь смогут только все участники, собравшись вместе.

Перейдем к формальным описаниям. Пусть имеется n участников разделения секрета, разрешенным множеством или разрешенной коалицией участников A будет называться множество участников, которые, объединившись, могут получить доступ к секрету, причем количество участников в таком множестве должно быть больше одного. Множества участников, которые при объединении, не смогут получить секретные данные - запрещенные коалиции (множества). Структурой доступа схемы разделения секрета будет называться пара (Δ, Г), где множество разрешенных множеств Г, а Δ - множество запрещенных коалиций[5].

Одним из действующих лиц в схеме разделения секрета является дилер. Задачей дилера является вычисление долей секрета и распределение их между участниками. Обозначим за S_0 конечное непустое множество значений секрета с соответствующей случайной величиной η, принимающей значение на декартовом произведении множеств $S_1 \times ... \times S_n$ и с распределением P на нем, где множества S_i конечны, η_i - соответствующие случайные величины, а s_i - значение η_i. Значение $(\eta_1, ..., \eta_n)$ дилер будет использовать в качестве набора долей секрета s_0. После выбора секрета s_0 с вероятностью $p(s_0)$ дилер отправляет участникам доли секрета $s_1, ..., s_n$ с вероятностью $P_{s_0}(s_1, ..., s_n)$, а именно для i-ого участника долей секрета будет являться s_i. Тогда коалиция

участников A получает набор $(s_i \mid i \in A)$. Чтобы схема разделения секрета реализовывала структуру доступа (Δ, Γ) мы должны обеспечить условие, чтобы все разрешенные коалиции могли восстановить секрет. Формально это можно записать так:

$$P(\eta = c_0 \mid \eta_i = c_i, i \in A) \in \{0, 1\} \text{ для } \forall A \in \Gamma$$

Отметим, что каждый из участников получает свою долю и не имеет информации о значениях других долей, однако знает все множества S_i , а также оба распределения вероятностей $p(s_0)$ и $P_{s_0}(s_1, ..., s_n)$.

Введем понятия совершенных схем разделения секрета. Совершенной схемой разделения секрета называется такая схема, в которой неразрешенные множества не получают никакой дополнительной информации к имеющейся априорной о возможном значении секрета. Это можно формализовать следующим образом:

$$P(\eta = c_0 \mid \eta_i = c_i, i \in A) = P(\eta = c_0) \text{ для } \forall A \in \Delta$$

Рассмотрим класс совершенных схем, а именно пороговые схемы. Будем называть (k,n)-пороговой схемой такую схему разделения секрета между n участниками, в которой любая группа из k и более участников разрешенная. К таким схемам относятся, например, схема Шамира[1] и схема Блэкли[2]. Такие схемы разделения секрета применяются для построения пороговых криптосистем. В пороговой криптосистеме сообщение может быть расшифровано определенной коалицией участников, между которыми и разделен секрет. Группа участников имеет общий открытый ключ шифрования, а ключ расшифрования разделен между ними с помощью схемы. Частным случаем такой системы является схема пороговой подписи. Пороговая криптография применяется для хранения секретного ключа, например, в государственной и военной сфере, а также она находит применение в облачных средах и схемах электронного голосования.

Но на практике пороговых схем бывает недостаточно, так как разрешенные множества могут быть произвольными. Одним из решений является выдача одному участнику нескольких ключей, но такое решение неэффективно. В 2010 году А.Абрамовым было предложено построение системы разделения секрета общего вида[3], основанная на кодах, исправляющих ошибки, в которой структура доступа может быть произвольной, при этом каждому из участников выдается лишь один ключ.

3. Схема Шамира

В 1979 году Ади Шамир в своей работе[1] предложил совершенную (k,n)-пороговую схему, в основе которой лежит интерполяция многочлена с коэффициентами из заданного поля Галуа с p элементами - $GF(p)$.

3.1. Описание схемы Шамира

Схема состоит из двух фаз. Во время первой фазы дилер генерирует многочлен F степени (k-1), генерируя случайным образом (k-1) элементов из GF(p). Эти элементы будут являться коэффициентами многочлена f_j, где $j \in (1, ..., k-1)$. Коэффициентом f_0 становится значение секрета s_0.

Во время второй фазы каждому из n участников сопоставляется ненулевой номер, и дилер отправляет “тень”, долю секрета, которая представляет из себя пару (i, F(i)), где i - порядковый номер участника, а F(i) - значение многочлена в этой точке.

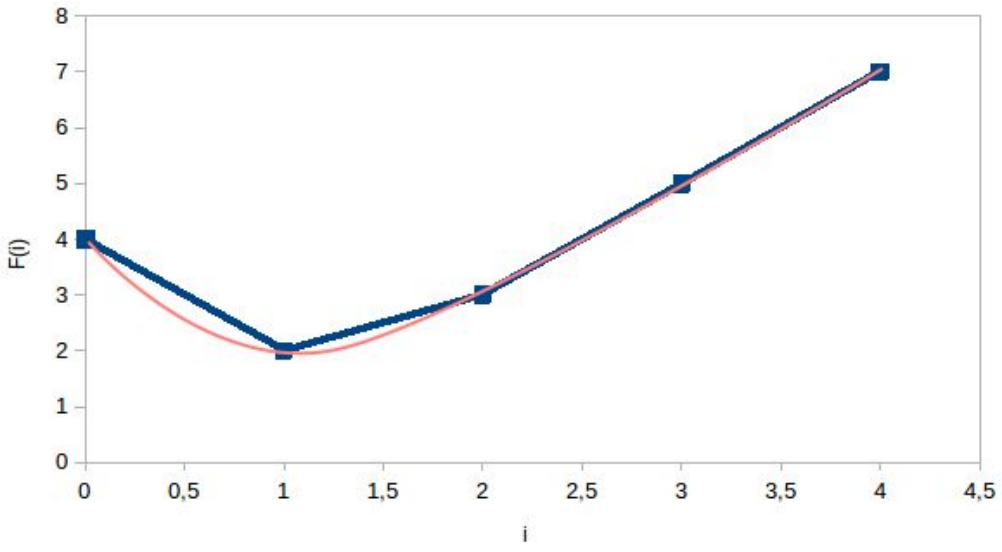

Рис.1: Пример схемы Шамира

Любой многочлен (k-1) степени можно однозначно восстановить по любым k различным точкам с помощью интерполяции. Для этого можно использовать, например, интерполяционную формулу Лагранжа. Таким образом любые k и более участников смогут восстановить многочлен F и вычислить значение F в точке 0, что будет являться значением секрета.

Очевидно, что эта схема совершенна, так как любые k и более участников однозначно восстанавливают секрет, а любые группы из менее k участников не получают никакой дополнительной информации о секрете.

3.2. Пример работы схемы

Пусть имеется (4, 6)-пороговая схема., то есть имеется 4 участника и любые 2 из них могут восстановить секрет. Зададим поле $GF(13)$ и секретный ключ s = 8. Во время первой фазы дилер выбирает многочлен F степени 3 со свободным коэффициентом, равным значению секрета: $4x^3 + 2x^2 + 8 \bmod 13$. После выбора многочлена генерируются 6 “теней” (x, F(x)): (1, 1), (2, 9), (3, 4), (4,10), (5,12), (6, 8), и затем они отправляются участникам.

Теперь любые 4 участника смогут восстановить многочлен, а значит и вычислить секрет. Предположим, восстановить секрет решили первые четыре участника. По формуле Лагранжа:

$$F_{k-1}(\mathrm{x}) = \sum_{j=1}^{k} f(x_j) \frac{\prod_{i \neq j} (x - x_i)}{\prod_{i \neq j} (x_j - x_i)},$$

где x_i - первая компонента “тени”, $f(x_i)$ - вторая компонента “тени” восстанавливается многочлен.

$$\sum_{j=1}^{k} f(x_j) \frac{\prod_{i \neq j} (x - x_i)}{\prod_{i \neq j} (x_j - x_i)} = 7 \cdot \frac{(x-2)(x-3)(x-4)}{(1-2)(1-3)(1-4)} + 9 \cdot \frac{(x-1)(x-3)(x-4)}{(2-1)(2-3)(2-4)} +$$

$$+ \; 4 \cdot \frac{(x-1)(x-2)(x-4)}{(3-1)(3-2)(3-4)} + + 10 \cdot \frac{(x-1)(x-2)(x-3)}{(4-1)(4-2)(3)} = 4x^3 + 2x^2 + 8$$

При вычислении многочлена в точке 0 участники получат значение секрета s.

3.3. Достоинства и недостатки схемы Шамира

Плюсы:

1. Масштабируемость. Количество участников можно увеличить до порядка поля p, при этом размер коалиции, способной восстановить секрет не меняется.
2. Идеальность. Размер каждой из “теней” равен размеру секрета.
3. Динамичность. Меняя используемый многочлен и пересчитывая “тени” и сохраняя секрет неизменным, вероятность нарушения защиты путем утечки “теней” уменьшается, так как для получения секрета нужно k или более “теней”, полученных на одном многочлене.
4. Совершенность. Участники, владеющие вместе менее k “тенями”, ничего не узнают о секрете.

Минусы:

1. Дилер-противник. В рассмотренной схеме мы не учитывали возможность того, что дилер может выдать неверные проекции.
2. Ненадежность дилера. Дилер может саботировать восстановление секрета.

3.4. Сложность вскрытия схемы

Нахождение свободного члена в конечном поле порядка p займет время порядка O(p), так как будет осуществлен полный перебор. Напомним, что секрет

выбирается без каких-либо определенных свойств, таким образом события, что секрет окажется i-тым ($i \in (0, ..., p-1)$ элементом поля равновероятны. Следовательно, если выбрать p достаточно большим, то перебор за разумное время становится невозможным, а вероятность угадать очень мала.

3.5. Применение

Схема Шамира применяется также для иерархических структур доступа. Такие структуры представляют из себя деревья, где каждый узел имеет доступ к меньшему объему информации, чем его родитель. Корень дерева имеет доступ ко всей информации. Один вариант реализации Шамир представил в своей работе[1], также над этой темой работал Котари, разработав схему, являющуюся обобщением нескольких[6]. Robert McEliece и Dilip Sarwate расширили схему Шамира с помощью кодов, исправляющих ошибки[7], а именно с помощью кода Рида-Соломона[4].

4. Обзор имеющихся решений

На данный момент не существует какой-либо стандартной библиотеки, реализующей схему Шамира. Ниже приведены две реализации схемы Шамира в открытом доступе:

1. B. Poettering, библиотека написана в 2006 году на языке C[8]

2. Daniel Silverstone, библиотека также написана в 2006 году на языке C[9]

5. Реализация системы

В качестве языка реализации был выбран язык C++. Выбор был сделан по нескольким причинам:

1. На настоящий момент для языка C++ нет библиотеки для схемы Шамира;

2. C++ с библиотекой NTL удобен в работе с большими числами, что особенно необходимо, чтобы сделать полный перебор возможного значения секрета невозможным.

3. C++ является объектно-ориентированным языком, что удобно для реализации конечного поля.

5.1. Класс Field

В классе Field реализовано поле Галуа простого порядка p - GF(p), то есть реализованы сложение, вычитание, деление и умножение элементов поля.

Оператор “+” реализует сложение по модулю p.

Оператор “ -” реализует вычитание по модулю p.

Оператор “*” реализует умножение по модулю p.

Оператор “/” реализует деление по модулю p.

Деление реализовано при помощи вспомогательной функции gsd, реализующей алгоритм Евклида для нахождения наибольшего общего делителя двух элементов. Данная функция принимает два элемента и возвращает элемент поля, являющийся их наибольшим общим делителем.

Также в функции division реализован еще один вариант деления, в котором

используется ассоциативный контейнер map. Выражение a/b есть не что иное, как $a * (1/b) = a * b^{-1}$, то есть элемент a умножается на элемент обратный b. В контейнер в качестве ключей помещены все элементы поля, значением для ключа является его обратный элемент в поле, вычисленный заранее. Данная функция использовалась для ускорения работы при небольших значениях порядка поля.

5.2. Компонента Dealer

Функция createPolynom принимает параметр порога k - минимальное количество участников, которые смогут восстановить секрет. На выходе функция выдает коэффициенты полинома, необходимого для создания “теней” участников.

Функция createShadows генерирует “тени” для всех n участников на основе принятого полинома. Для этого он генерирует n случайных элементов заданного поля и вычисляет значение полинома в этих точках. Как уже было сказано, “тень”, полученная каждым участником является парой - элемент выбранного конечного поля и значения полинома в этой точке.

5.3. Компонента SecretRecovery

Функция LagrangeInterpolation восстанавливает полином по k принятым от участников “теням”с помощью интерполяционной формулы Лагранжа:

$$F_{k-1}(\mathrm{x}) = \sum_{j=1}^{k} f(x_j) \frac{\prod_{i \neq j} (x - x_i)}{\prod_{i \neq j} (x_j - x_i)},$$

где x_i - первая компонента “тени”, $f(x_i)$ - вторая компонента. Если было принято более k “теней”, полином восстанавливает по первым k полученным.

Функция findSecret восстанавливает секрет. Она принимает коэффициенты полинома и вычисляет его значение в точке 0. Как было сказано выше, значение полинома в точке 0 будет являться значением секрета.

5.4. Оптимизация

Для ускорения работы программы был использован механизм для распараллеливания OpenMP. Он был использован в генерации полинома, создания “теней” и в восстановлении секрета. Сравнение времени работы программы с использованием данного механизма и без него будут представлены ниже.

6. Характеристики работы библиотеки

Были проведены замеры времени работы библиотеки с разным значением порядка *p* поля GF(*p*). Ниже приведен график получившихся результатов для (k, n) = (10, 20), где x-степень двойки для p. Так как p должно быть простым числом, к 2^x добавлялось число, не превышающее 2^5.

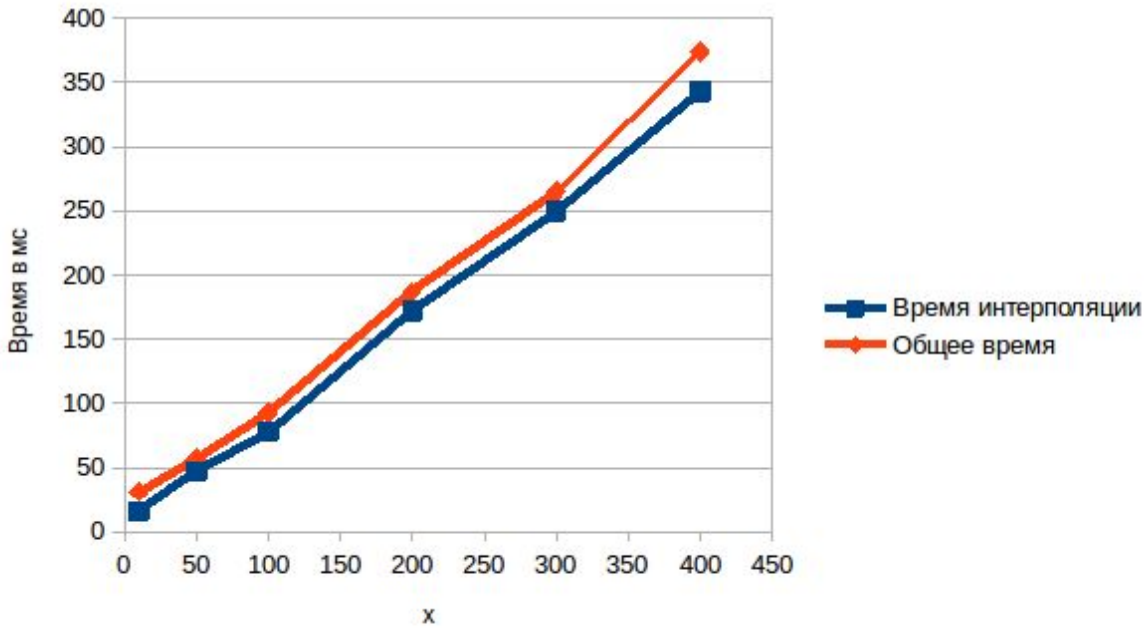

Рис. 2: График зависимости времени от числа элементов в поле

Также были проведены замеры времени при подключении механизма OpenMP. Программа с 2-мя потоками работает быстрее приблизительно в 1.8 раз. Стоит уточнить, что программы запускались на одном устройстве при равных условиях.

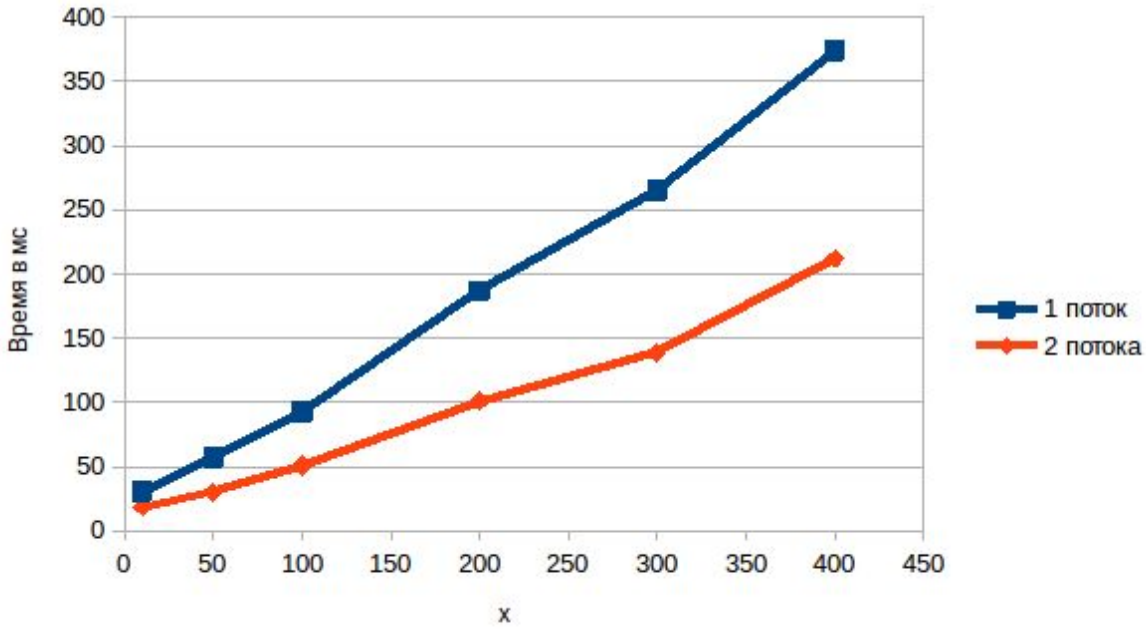

Рис. 3: Время работы для 1-ого и 2-ух потоков

7. Заключение

Результатом данной работы являются:

1. Реализация библиотеки схемы Шамира на языке C++;
2. Тестирование библиотеки.

Список литературы

[1] Adi Shamir. How to share a secret. - 1979. - Communications of the ACM 22 (11): 612–613

[2] Blakley, G. R. Safeguarding cryptographic keys. — 1979. — Proceedings of the National Computer Conference 48: 313–317.

[3] Абрамов А.П. Системы разделения секрета общего доступа. — 2010. — 13 с.

[4] Блейхут Р. Теория и практика кодов, контролирующих ошибки//Theory and Practice of Error Control Codes. — М.: Мир, 1986. — 576

[5] Под общ. ред. Ященко В.В. Введение в криптографию. — 2-е изд., испр. — М.: МЦНМО: «ЧеРо», 1999. — 272 с.

[6]Kothari, S. C., Generalized linear threshold scheme. Advances in Cryptology - CRYPTO 84, LNCS 196, 231 - 241, 1985.

[7]McEliece, R. J., Sarwate, D. V., On sharing secrets and Reed Solomon codes. Comm. of ACM, 24, 583 - 584, 1981.

[8] B. Poettering - 2006 - http://point-at-infinity.org/ssss/

[9]Daniel Silverstone - 2006 - http://www.digital-scurf.org/software/libgfshare